National University
of computer and emerging sciences

Project: AI Chatbot for Phishing Email Detection and Generation

Team Members:

• Hamza Yousuf (22k4748)

• Sohaib Jaber (22k4751)

1. Motivation

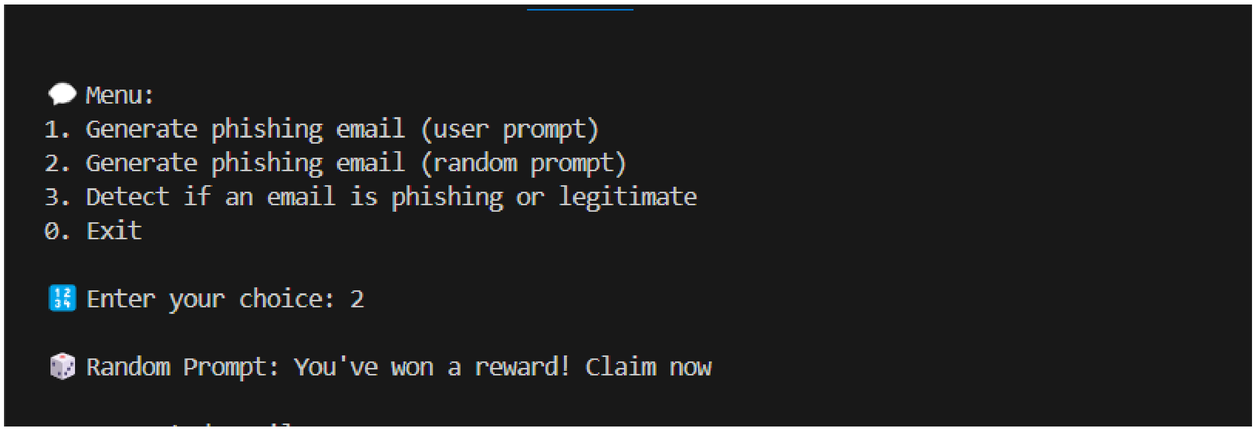
In the digital age, email communication has become an essential part of personal, academic, and organizational interaction. However, with this widespread usage comes an increased risk of cyber threats, particularly phishing attacks. These deceptive emails are crafted to manipulate users into revealing sensitive information such as passwords, credit card numbers, or personal data. The growing sophistication of such threats highlights the need for intelligent systems that can both identify and illustrate phishing behavior. Motivated by this challenge, we aimed to develop an AI-based chatbot that not only detects phishing emails using a trained BERT model but also generates realistic phishing examples using a fine-tuned GPT-2 model. Our objective was to create a comprehensive educational and security-oriented tool that bridges the gap between awareness and prevention, making users more resilient against phishing attempts.

2. Overview

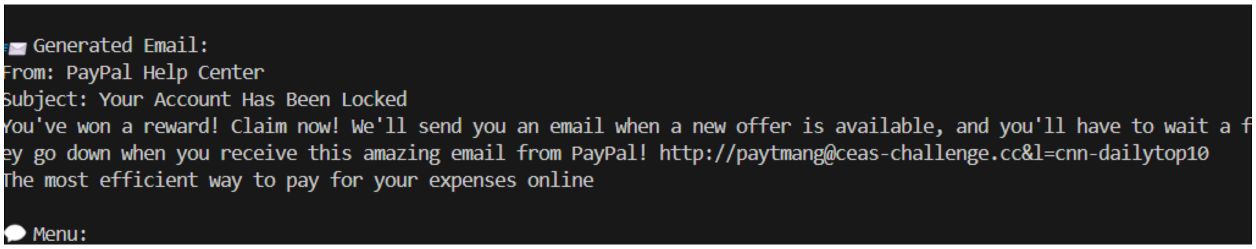
2.1 Significance of the Project

Phishing continues to be one of the most dangerous and widespread forms of social engineering attacks, compromising personal and organizational data daily. The proposed project tackles this threat by offering a dual-functionality AI chatbot capable of both generating and detecting phishing emails using state-of-the-art language models.

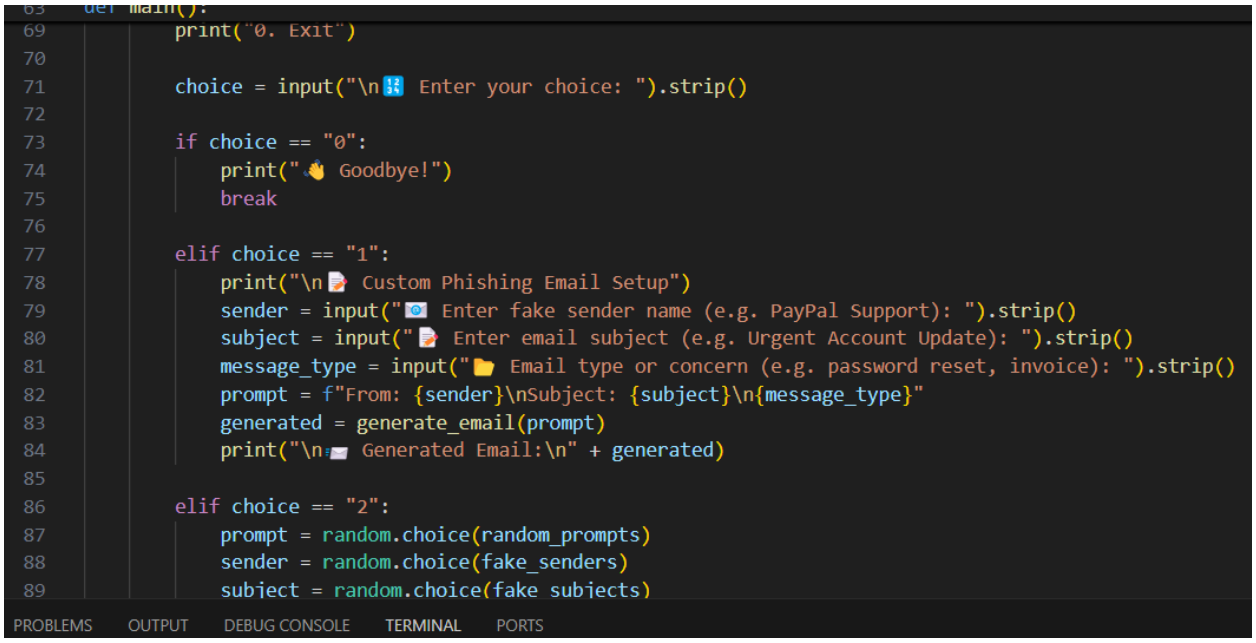
This project holds practical value in cybersecurity education, awareness training, and penetration testing. It serves as a simulation environment for adversarial testing and enhances understanding of phishing tactics by demonstrating how malicious emails are crafted and detected. The use of machine learning adds academic depth, while the chatbot interface ensures user engagement and accessibility.



```
💬 Menu:
1. Generate phishing email (user prompt)
2. Generate phishing email (random prompt)
3. Detect if an email is phishing or legitimate
0. Exit

🔢 Enter your choice: 2

🎲 Random Prompt: You've won a reward! Claim now
```



```
📨 Generated Email:
From: PayPal Help Center
Subject: Your Account Has Been Locked
You've won a reward! Claim now! We'll send you an email when a new offer is available, and you'll have to wait a f
ey go down when you receive this amazing email from PayPal! http://paytmang@ceas-challenge.cc&l=cnn-dailytop10
The most efficient way to pay for your expenses online

💬 Menu:
```

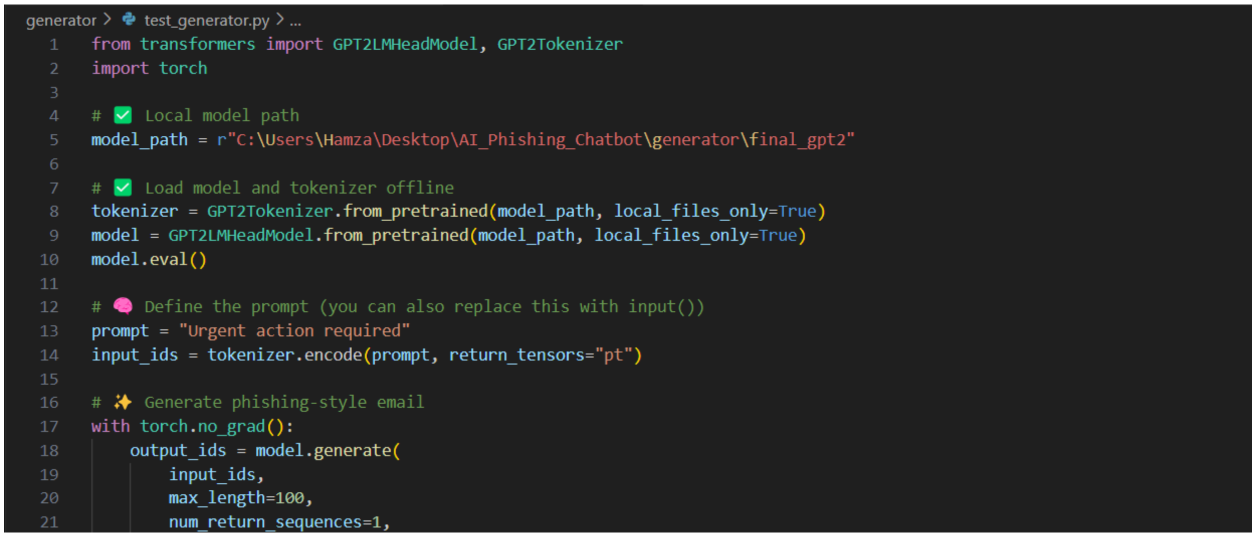


```
def main():
        print("0. Exit")

        choice = input("\n🔢 Enter your choice: ").strip()

        if choice == "0":
            print("👋 Goodbye!")
            break

        elif choice == "1":
            print("\n📝 Custom Phishing Email Setup")
            sender = input("📧 Enter fake sender name (e.g. PayPal Support): ").strip()
            subject = input("📝 Enter email subject (e.g. Urgent Account Update): ").strip()
            message_type = input("📁 Email type or concern (e.g. password reset, invoice): ").strip()
            prompt = f"From: {sender}\nSubject: {subject}\n{message_type}"
            generated = generate_email(prompt)
            print("\n📨 Generated Email:\n" + generated)

        elif choice == "2":
            prompt = random.choice(random_prompts)
            sender = random.choice(fake_senders)
            subject = random.choice(fake_subjects)
```

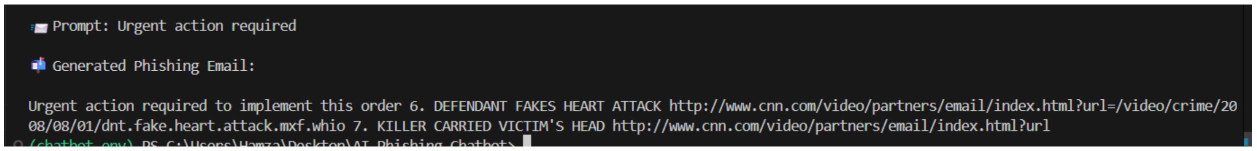
2.2 Description of the Project

This AI-powered chatbot integrates two major models:

- **GPT-2 (fine-tuned)** for phishing email generation.

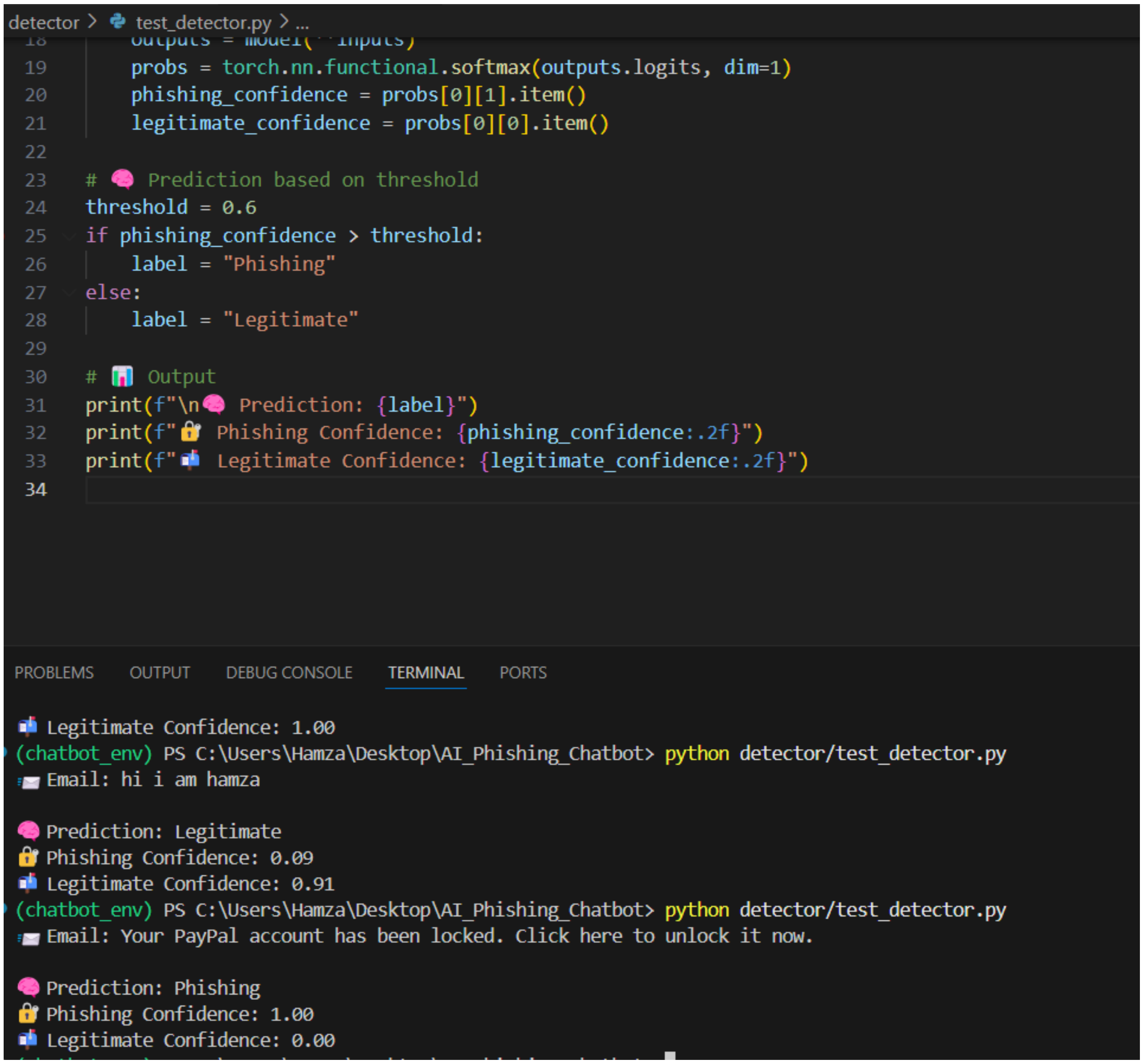


```
from transformers import GPT2LMHeadModel, GPT2Tokenizer
import torch

# ✅ Local model path
model_path = r"C:\Users\Hamza\Desktop\AI_Phishing_Chatbot\generator\final_gpt2"

# ✅ Load model and tokenizer offline
tokenizer = GPT2Tokenizer.from_pretrained(model_path, local_files_only=True)
model = GPT2LMHeadModel.from_pretrained(model_path, local_files_only=True)
model.eval()

# 🧠 Define the prompt (you can also replace this with input())
prompt = "Urgent action required"
input_ids = tokenizer.encode(prompt, return_tensors="pt")

# ✨ Generate phishing-style email
with torch.no_grad():
    output_ids = model.generate(
        input_ids,
        max_length=100,
        num_return_sequences=1,
```



```
📨 Prompt: Urgent action required

📬 Generated Phishing Email:

Urgent action required to implement this order 6. DEFENDANT FAKES HEART ATTACK http://www.cnn.com/video/partners/email/index.html?url=/video/crime/20
08/08/01/dnt.fake.heart.attack.mxf.whio 7. KILLER CARRIED VICTIM'S HEAD http://www.cnn.com/video/partners/email/index.html?url
(chatbot_env) PS C:\Users\Hamza\Desktop\AI_Phishing_Chatbot>
```

- **BERT (fine-tuned)** for phishing email detection.



detector > test_detector.py > ...

```python
    outputs = model(**inputs)
    probs = torch.nn.functional.softmax(outputs.logits, dim=1)
    phishing_confidence = probs[0][1].item()
    legitimate_confidence = probs[0][0].item()

# 🧠 Prediction based on threshold
threshold = 0.6
if phishing_confidence > threshold:
    label = "Phishing"
else:
    label = "Legitimate"

# 📊 Output
print(f"\n🧠 Prediction: {label}")
print(f"🔐 Phishing Confidence: {phishing_confidence:.2f}")
print(f"📬 Legitimate Confidence: {legitimate_confidence:.2f}")

```

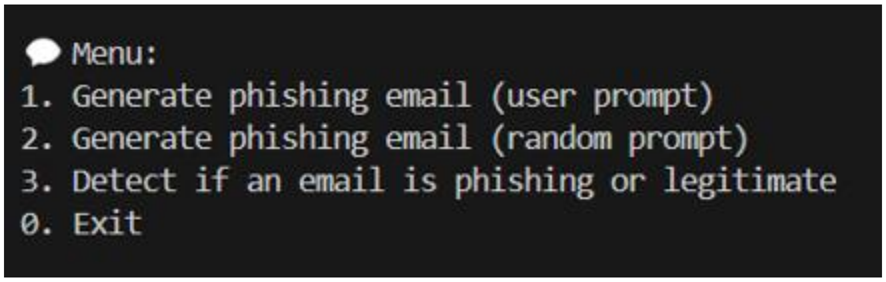
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
📬 Legitimate Confidence: 1.00
(chatbot_env) PS C:\Users\Hamza\Desktop\AI_Phishing_Chatbot> python detector/test_detector.py
📨 Email: hi i am hamza

🧠 Prediction: Legitimate
🔐 Phishing Confidence: 0.09
📬 Legitimate Confidence: 0.91
(chatbot_env) PS C:\Users\Hamza\Desktop\AI_Phishing_Chatbot> python detector/test_detector.py
📨 Email: Your PayPal account has been locked. Click here to unlock it now.

🧠 Prediction: Phishing
🔐 Phishing Confidence: 1.00
📬 Legitimate Confidence: 0.00
```

The system offers four core options:

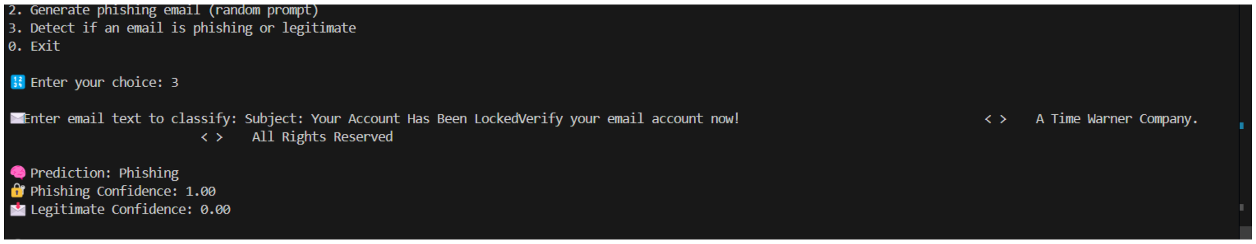
1. Generate phishing email using a user-defined prompt.
2. Generate phishing email randomly using predefined subjects/senders.
3. Detect if a given email is phishing or legitimate.
4. Exit the chatbot.

Both models are loaded from local directories and do not require internet access. The generation process includes user-controlled components such as sender name, subject line, and message concern, and filters out unwanted content like repeated CNN links. For detection, BERT classifies input text and displays phishing vs. legitimate confidence scores.



```
💬 Menu:
1. Generate phishing email (user prompt)
2. Generate phishing email (random prompt)
3. Detect if an email is phishing or legitimate
0. Exit
```

The detector detecting the mail provided to it



```
2. Generate phishing email (random prompt)
3. Detect if an email is phishing or legitimate
0. Exit

🔢 Enter your choice: 3

✉️Enter email text to classify: Subject: Your Account Has Been LockedVerify your email account now!                        < >    A Time Warner Company.
                       < >    All Rights Reserved

🧠 Prediction: Phishing
🔐 Phishing Confidence: 1.00
📩 Legitimate Confidence: 0.00
```

2.4 Project Category

This is a **Product-Based Project** that simulates real-world phishing scenarios using generative AI and evaluates them with a detection engine, allowing both offensive (red team) and defensive (blue team) perspectives.

3. Features / Scope / Modules

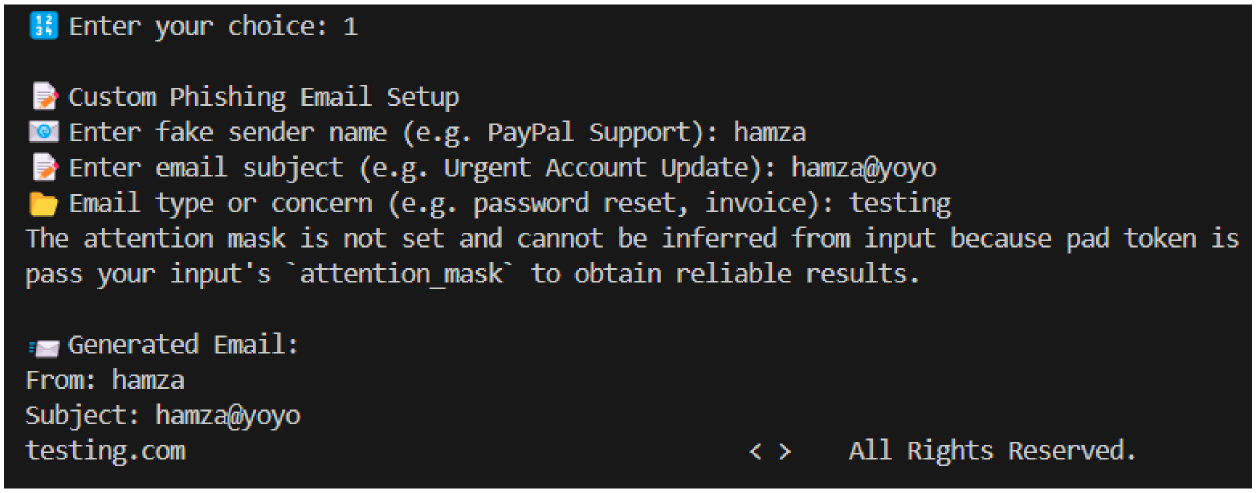
This project provides a dual-functionality interface where users can simulate and detect phishing emails. Below are the key features and modules:

---

1. AI-Based Phishing Email Generation

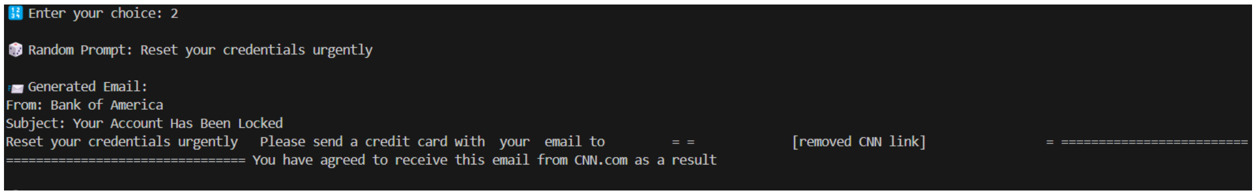
The system uses a fine-tuned **GPT-2** model to generate phishing-style emails. It supports two modes:

- **User Prompt Mode**: The user defines sender name, subject, and the type of phishing content (e.g., password reset, invoice).



```
🔢 Enter your choice: 1

📝 Custom Phishing Email Setup
📧 Enter fake sender name (e.g. PayPal Support): hamza
📝 Enter email subject (e.g. Urgent Account Update): hamza@yoyo
📁 Email type or concern (e.g. password reset, invoice): testing
The attention mask is not set and cannot be inferred from input because pad token is
pass your input's `attention_mask` to obtain reliable results.

📨 Generated Email:
From: hamza
Subject: hamza@yoyo
testing.com                                        < >    All Rights Reserved.
```

- **Random Prompt Mode**: The system auto-generates a phishing email using random fake sender names, subjects, and message concerns.



```
🔢 Enter your choice: 2

🎲 Random Prompt: Reset your credentials urgently

📨 Generated Email:
From: Bank of America
Subject: Your Account Has Been Locked
Reset your credentials urgently   Please send a credit card with  your  email to        = =            [removed CNN link]              = ========================
================================ You have agreed to receive this email from CNN.com as a result
```

These generated emails mimic real-world phishing messages and help in understanding adversarial techniques.

---

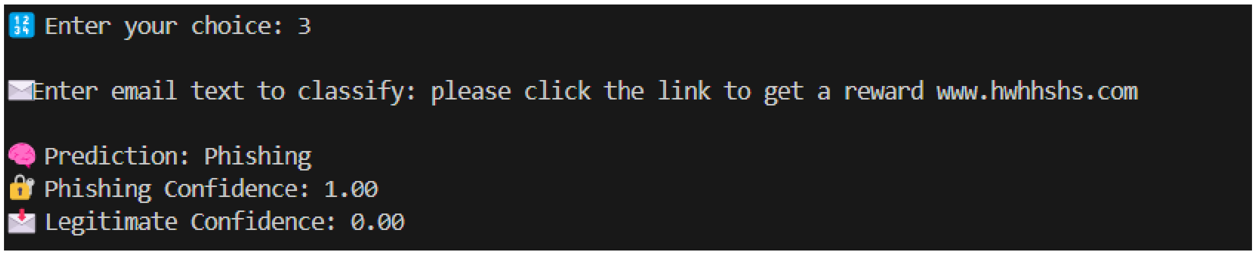
2. Real-Time Phishing Email Detection using BERT

The system includes a **fine-tuned BERT-based detector** trained on a cleaned and balanced dataset of phishing and legitimate emails. Users can input any email message, and the detector will instantly:

- Classify it as **Phishing** or **Legitimate**
- Show the **confidence scores** for both classes

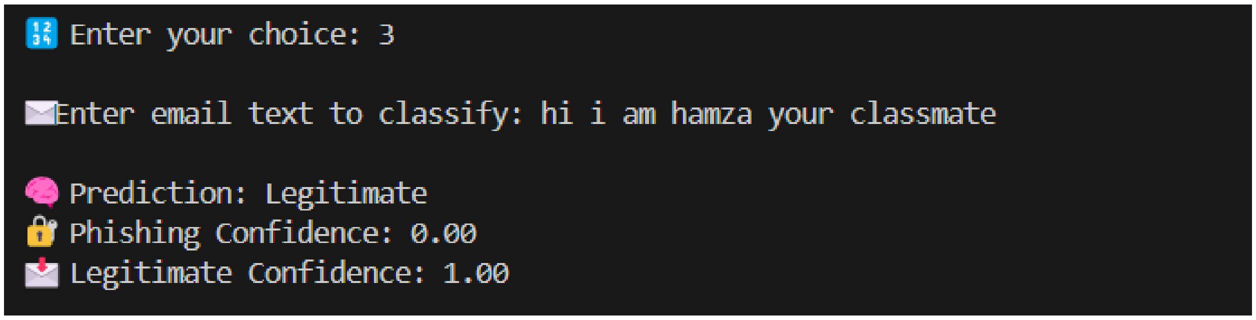
This module is essential for testing the effectiveness of phishing campaigns and training detection models.

The figure shows a phishing mail detection from the Bert model



```
🔢 Enter your choice: 3

✉️Enter email text to classify: please click the link to get a reward www.hwhhshs.com

🧠 Prediction: Phishing
🔐 Phishing Confidence: 1.00
📩 Legitimate Confidence: 0.00
```

The figure shows a legitimate mail detection from the Bert model



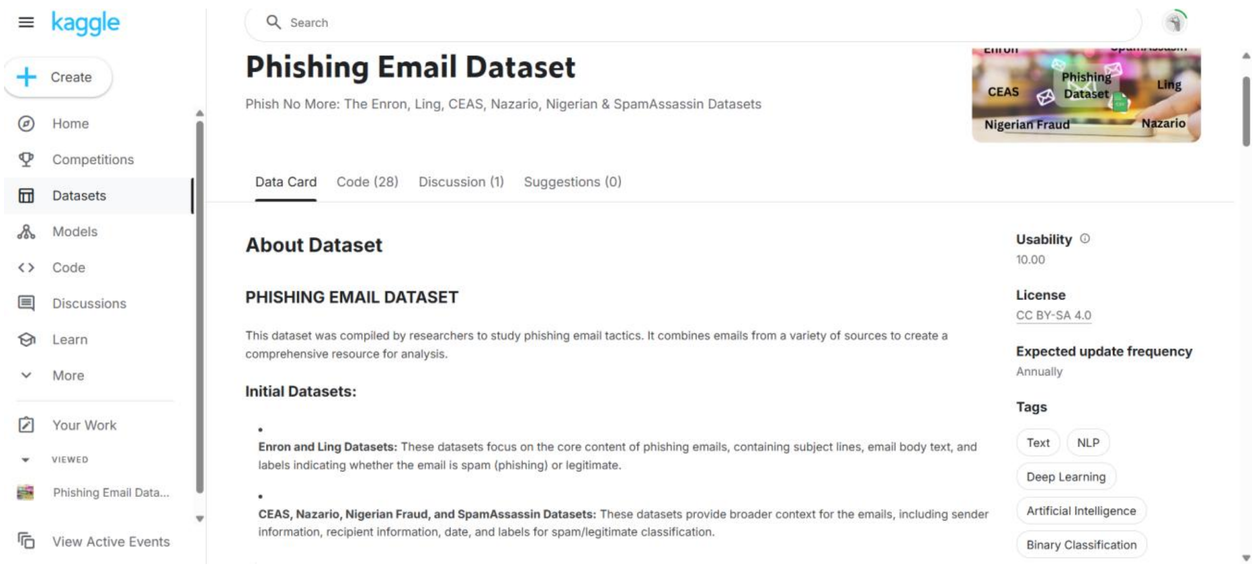
```
🔢 Enter your choice: 3

✉️Enter email text to classify: hi i am hamza your classmate

🧠 Prediction: Legitimate
🔐 Phishing Confidence: 0.00
📩 Legitimate Confidence: 1.00
```

3. Email Dataset Cleaning and Balancing Pipeline

Before training, a data processing pipeline was implemented to:

- Merge the **subject** and **body** fields into a single unified text
- **Remove noise** such as spam logs, antivirus reports, and irrelevant patterns (e.g., SVN commits, .cvd, .py references)
- Drop empty or duplicate entries
- **Balance the dataset** by sampling equal numbers of phishing and legitimate emails for fairness in model training

Source for dataset



Phishing Email Dataset

Phish No More: The Enron, Ling, CEAS, Nazario, Nigerian & SpamAssassin Datasets

About Dataset

PHISHING EMAIL DATASET

This dataset was compiled by researchers to study phishing email tactics. It combines emails from a variety of sources to create a comprehensive resource for analysis.

Initial Datasets:

- **Enron and Ling Datasets:** These datasets focus on the core content of phishing emails, containing subject lines, email body text, and labels indicating whether the email is spam (phishing) or legitimate.
- **CEAS, Nazario, Nigerian Fraud, and SpamAssassin Datasets:** These datasets provide broader context for the emails, including sender information, recipient information, date, and labels for spam/legitimate classification.

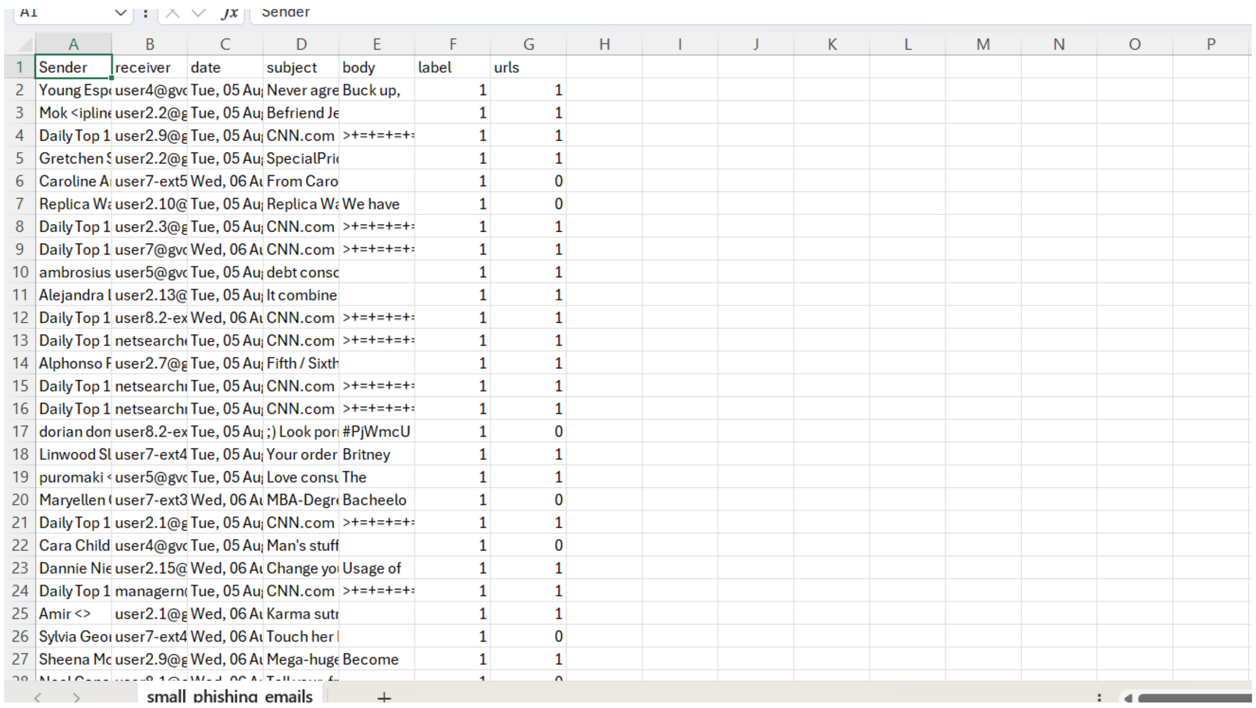
Usability
10.00

License
CC BY-SA 4.0

Expected update frequency
Annually

Tags
Text NLP Deep Learning Artificial Intelligence Binary Classification

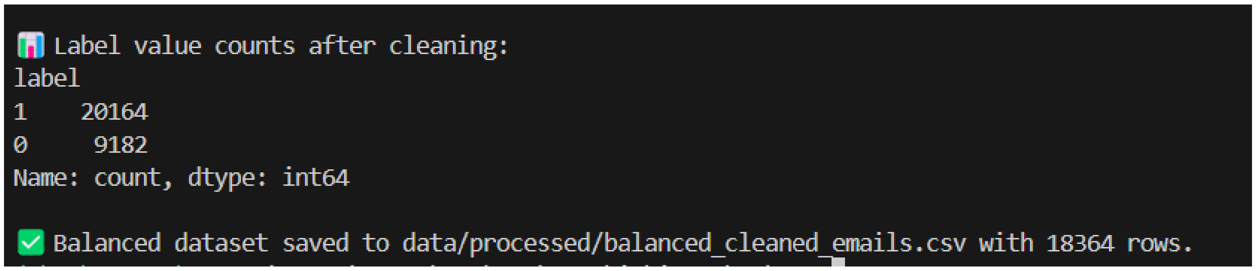
Original dataset downloaded



	Sender	receiver	date	subject	body	label	urls
2	Young Esp	user4@gv	Tue, 05 Au	Never agre	Buck up,	1	1
3	Mok <ipline	user2.2@g	Tue, 05 Au	Befriend Je		1	1
4	Daily Top 1	user2.9@g	Tue, 05 Au	CNN.com	>+=+=+=+:	1	1
5	Gretchen S	user2.2@g	Tue, 05 Au	SpecialPri		1	1
6	Caroline A	user7-ext5	Wed, 06 Au	From Caro		1	0
7	Replica Wa	user2.10@	Tue, 05 Au	Replica Wa	We have	1	0
8	Daily Top 1	user2.3@g	Tue, 05 Au	CNN.com	>+=+=+=+:	1	1
9	Daily Top 1	user7@gv	Wed, 06 Au	CNN.com	>+=+=+=+:	1	1
10	ambrosius	user5@gv	Tue, 05 Au	debt consc		1	1
11	Alejandra l	user2.13@	Tue, 05 Au	It combine		1	1
12	Daily Top 1	user8.2-ex	Wed, 06 Au	CNN.com	>+=+=+=+:	1	1
13	Daily Top 1	netsearche	Tue, 05 Au	CNN.com	>+=+=+=+:	1	1
14	Alphonso F	user2.7@g	Tue, 05 Au	Fifth / Sixth		1	1
15	Daily Top 1	netsearch	Tue, 05 Au	CNN.com	>+=+=+=+:	1	1
16	Daily Top 1	netsearch	Tue, 05 Au	CNN.com	>+=+=+=+:	1	1
17	dorian don	user8.2-ex	Tue, 05 Au	;) Look por	#PjWmcU	1	0
18	Linwood Sl	user7-ext4	Tue, 05 Au	Your order	Britney	1	1
19	puromaki <	user5@gv	Tue, 05 Au	Love consu	The	1	1
20	Maryellen (	user7-ext3	Wed, 06 Au	MBA-Degre	Bacheelo	1	0
21	Daily Top 1	user2.1@g	Tue, 05 Au	CNN.com	>+=+=+=+:	1	1
22	Cara Child	user4@gv	Tue, 05 Au	Man's stuff		1	0
23	Dannie Nie	user2.15@	Wed, 06 Au	Change yo	Usage of	1	1
24	Daily Top 1	managern	Tue, 05 Au	CNN.com	>+=+=+=+:	1	1
25	Amir <>	user2.1@g	Wed, 06 Au	Karma sutr		1	1
26	Sylvia Geo	user7-ext4	Wed, 06 Au	Touch her		1	0
27	Sheena Mc	user2.9@g	Wed, 06 Au	Mega-huge	Become	1	1

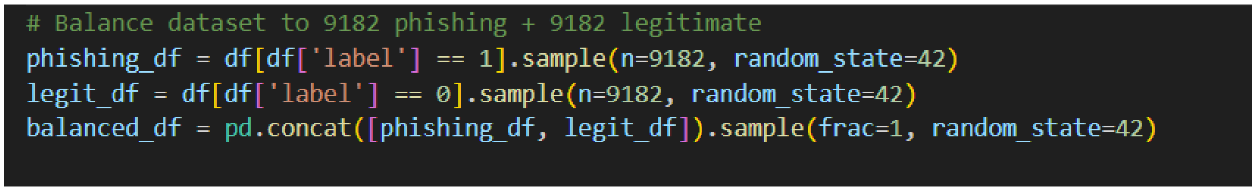
small_phishing_emails

After cleaning



```
📊 Label value counts after cleaning:
label
1    20164
0     9182
Name: count, dtype: int64

✅ Balanced dataset saved to data/processed/balanced_cleaned_emails.csv with 18364 rows.
```

When training the model the dataset was picked in equal number

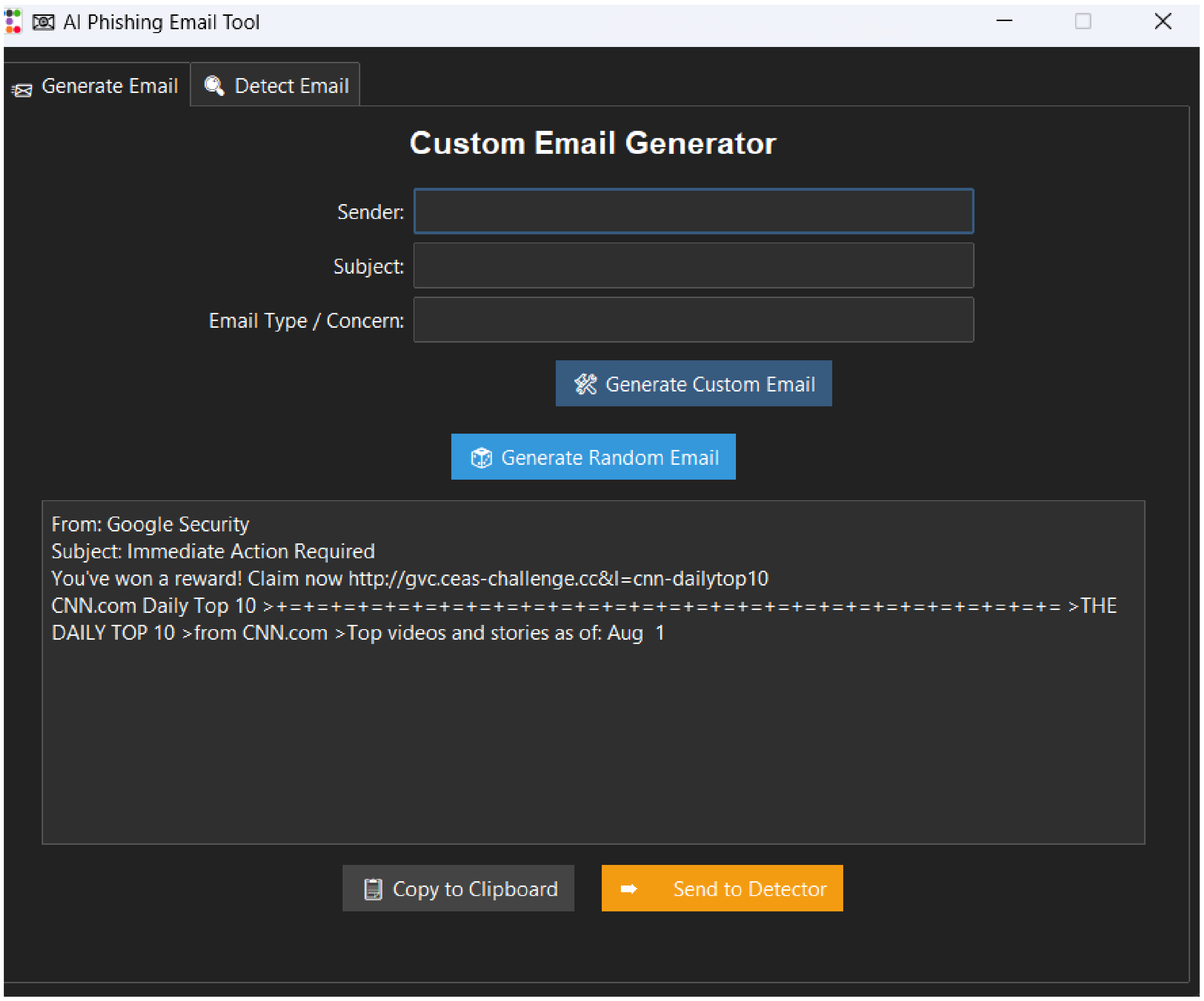


```
# Balance dataset to 9182 phishing + 9182 legitimate
phishing_df = df[df['label'] == 1].sample(n=9182, random_state=42)
legit_df = df[df['label'] == 0].sample(n=9182, random_state=42)
balanced_df = pd.concat([phishing_df, legit_df]).sample(frac=1, random_state=42)
```

4.Graphical User Interface (GUI) for User Interaction

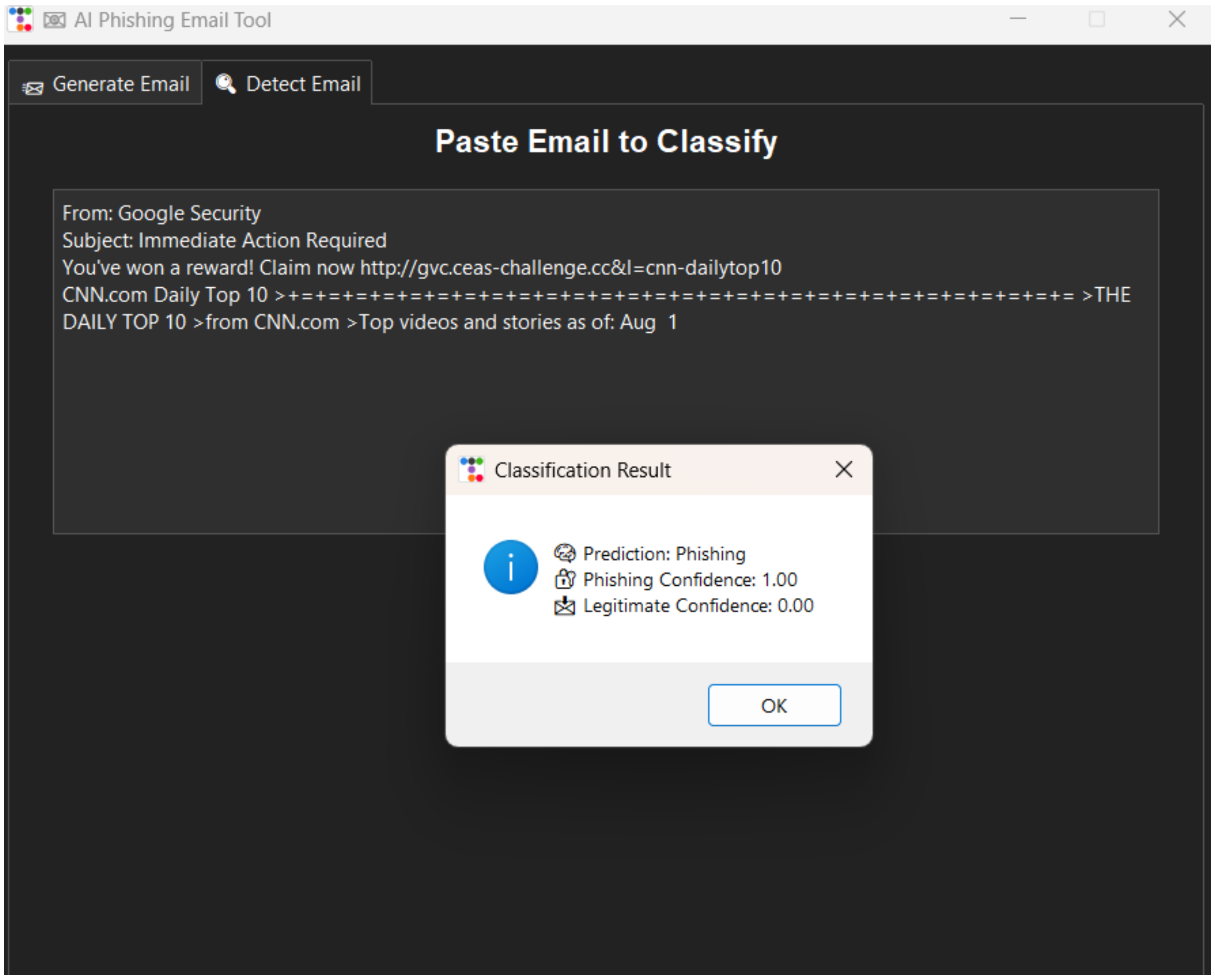
A clean and intuitive GUI was developed using ttkbootstrap and tkinter. It allows users to interact with the phishing email generator and detector models through:

- **Email Generator Tab**: Enables both custom and random phishing email generation with fields for sender, subject, and type of concern.



```
From: Google Security
Subject: Immediate Action Required
You've won a reward! Claim now http://gvc.ceas-challenge.cc&l=cnn-dailytop10
CNN.com Daily Top 10 >+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+= >THE
DAILY TOP 10 >from CNN.com >Top videos and stories as of: Aug  1
```

- **Email Detection Tab**: Allows users to paste any email content and get instant classification results.


AI Phishing Email Tool

Generate Email | Detect Email

Paste Email to Classify

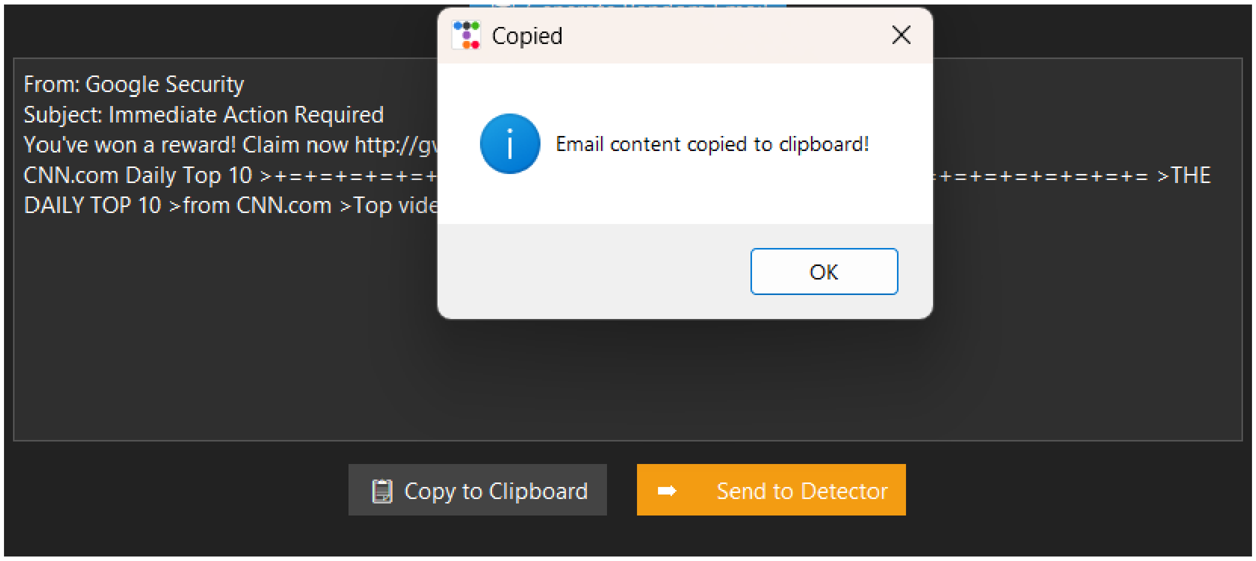
From: Google Security
Subject: Immediate Action Required
You've won a reward! Claim now http://gvc.ceas-challenge.cc&l=cnn-dailytop10
CNN.com Daily Top 10 >+= >THE DAILY TOP 10 >from CNN.com >Top videos and stories as of: Aug 1

Classification Result

Prediction: Phishing
Phishing Confidence: 1.00
Legitimate Confidence: 0.00

OK

- **Clipboard and Detector Integration**: Generated emails can be copied or directly sent to the detection module.


From: Google Security
Subject: Immediate Action Required
You've won a reward! Claim now http://gv
CNN.com Daily Top 10 >+=+=+=+=+=+ ... +=+=+=+=+=+=+=+= >THE
DAILY TOP 10 >from CNN.com >Top vide

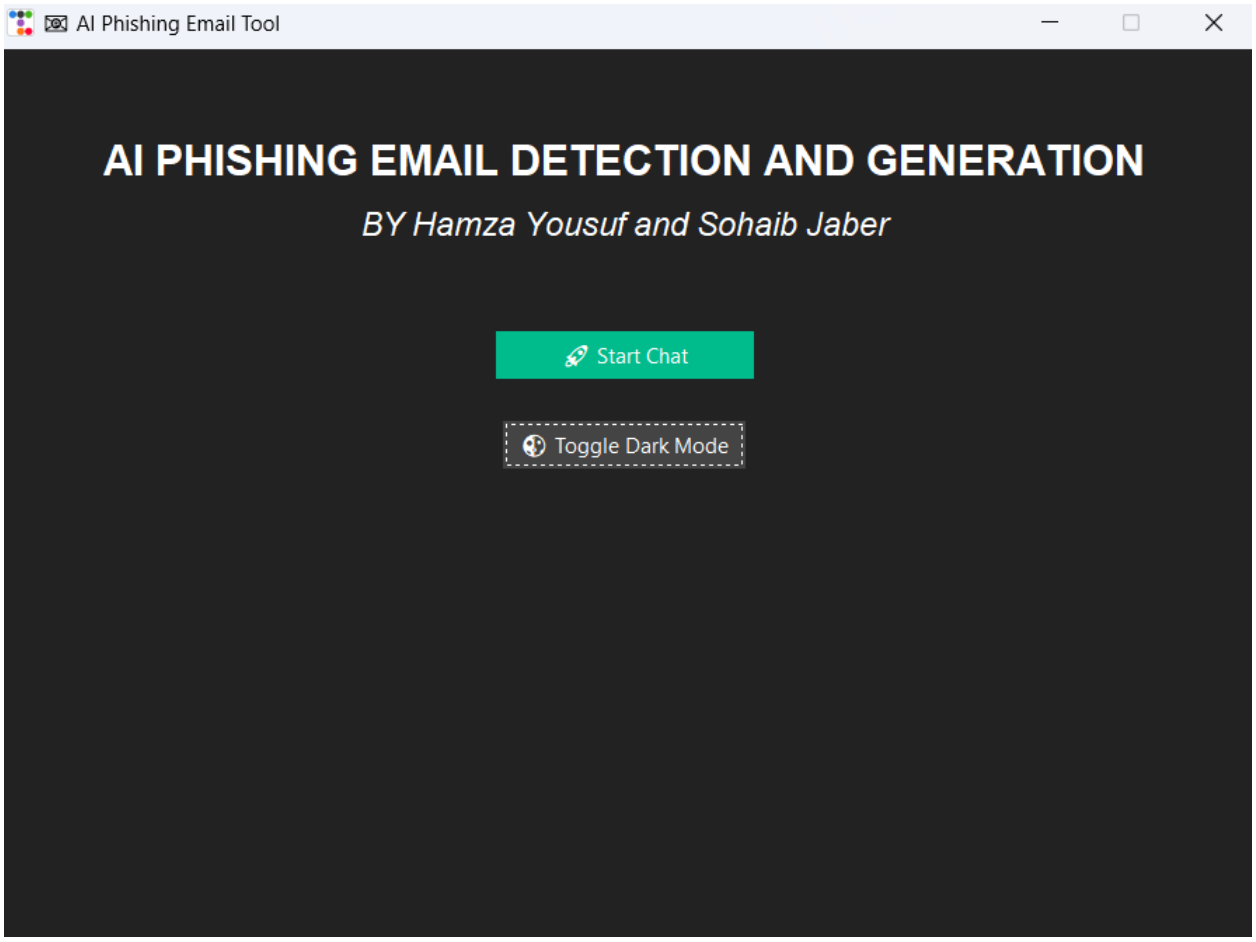
Copied

Email content copied to clipboard!

OK

Copy to Clipboard | Send to Detector

Main screen:



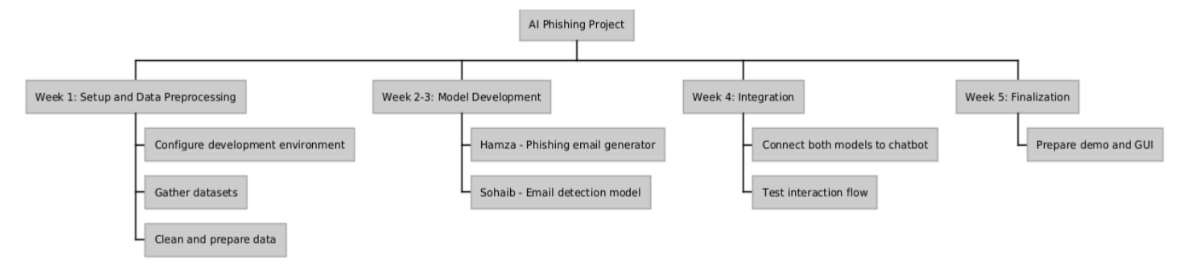
AI PHISHING EMAIL DETECTION AND GENERATION

BY Hamza Yousuf and Sohaib Jaber

4. Project Planning

The project was strategically divided into weekly milestones to ensure focused development and timely completion. Each team member had clearly defined responsibilities, allowing for parallel progress on different components. The detailed breakdown is as follows:

- **Week 1: Setup and Preparation**
  - Environment setup and installation of necessary libraries
  - Dataset collection (phishing and legitimate emails)
  - Initial data cleaning and exploration
- **Week 2–3: Model Development Phase**
  - *Hamza Yousuf*: Developed and trained the **GPT-2 phishing email generator**
  - *Sohaib Jaber*: Implemented and trained the **BERT-based phishing email detector**
  - Both models were tested on small batches to validate learning behavior
- **Week 4: Integration and Testing**
  - Integrated both models into a unified interface
  - Developed the logic for chatbot interaction flow
  - Conducted initial internal testing to verify full functionality
- **Week 5: Refinement and Finalization**
  - Enhanced UI through a GUI built with Tkinter and TTKBootstrap
  - Conducted full system demo preparation and testing


AI Phishing Project
Week 1: Setup and Data Preprocessing
Configure development environment
Gather datasets
Clean and prepare data
Week 2-3: Model Development
Hamza - Phishing email generator
Sohaib - Email detection model
Week 4: Integration
Connect both models to chatbot
Test interaction flow
Week 5: Finalization
Prepare demo and GUI

5. Project Feasibility

Technical Feasibility

The project is technically feasible, as it is built using well-supported open-source libraries such as Hugging Face's transformers, PyTorch, and ttkbootstrap for GUI development. All components (email generator, detector, and GUI) were tested successfully on a local machine without requiring cloud GPU resources. Models were trained and executed within the local environment using available datasets, ensuring full control and offline functionality.

Economic Feasibility

The entire system was developed using free and open-source tools. No commercial APIs or paid resources were involved. The only cost involved was time investment and system resources. The benefit of this project lies in its educational value and its potential to serve as a base for real-world phishing detection tools in academic or research contexts.

Schedule Feasibility

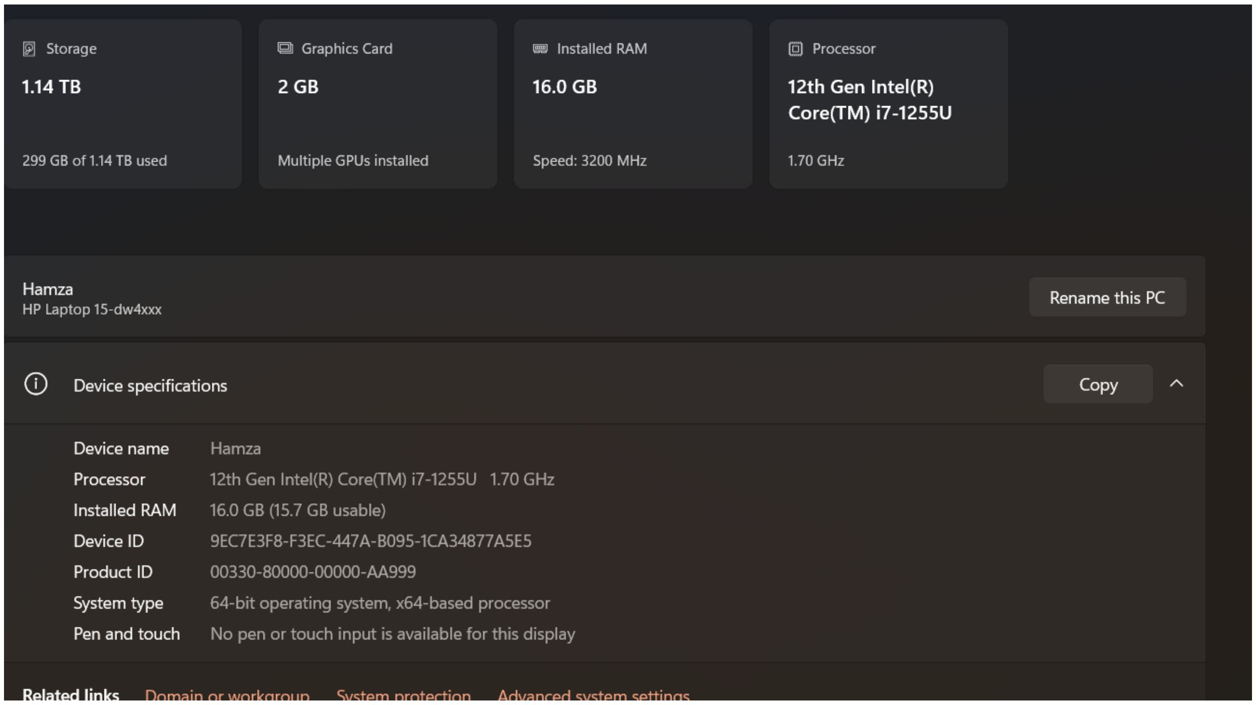
Despite the complexity, the project was completed within the proposed 5-week time frame. A structured development schedule helped divide the workload efficiently between the two team members. Model training and integration were prioritized early to allow sufficient time for GUI development, testing, and documentation in the final week.

6. Hardware and Software Requirements

Hardware Requirements

This project was executed on a system with the following specifications:

- **Device:** HP Laptop 15-dw4xxx
- **Processor:** 12th Gen Intel(R) Core(TM) i7-1255U @ 1.70 GHz
- **Installed RAM:** 16 GB (15.7 GB usable)
- **System Type:** 64-bit OS, x64-based processor
- **Storage:** 1.14 TB with 299 GB used
- **Graphics:** 2 GB Graphics Memory (Multiple GPUs)



Storage
1.14 TB
299 GB of 1.14 TB used

Graphics Card
2 GB
Multiple GPUs installed

Installed RAM
16.0 GB
Speed: 3200 MHz

Processor
12th Gen Intel(R) Core(TM) i7-1255U
1.70 GHz

Hamza
HP Laptop 15-dw4xxx

Device specifications

Device name	Hamza
Processor	12th Gen Intel(R) Core(TM) i7-1255U 1.70 GHz
Installed RAM	16.0 GB (15.7 GB usable)
Device ID	9EC7E3F8-F3EC-447A-B095-1CA34877A5E5
Product ID	00330-80000-00000-AA999
System type	64-bit operating system, x64-based processor
Pen and touch	No pen or touch input is available for this display

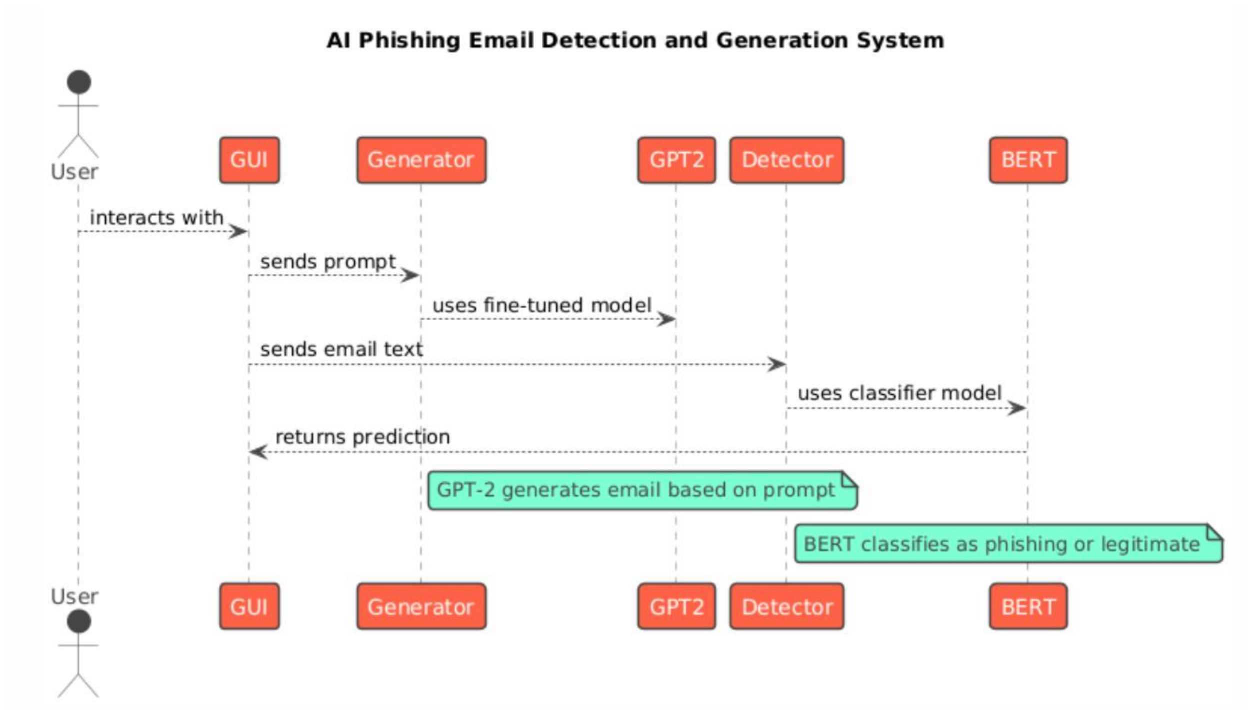
Software Requirements

- **Operating System:** Windows 11 (64-bit)
- **Python Version:** Python 3.10+
- **Libraries Used:**
  - transformers (for GPT-2 and BERT)
  - torch (PyTorch deep learning backend)
  - ttkbootstrap and tkinter (GUI interface)
  - pandas, regex, os (data handling and preprocessing)
- **IDE/Tool:** VS Code / Jupyter Notebook
- **Model Sources:**
  - GPT-2 (from Hugging Face Transformers)
  - BERT (pre-trained for binary classification)

  Even thou the system of this specification was used, the more important was the software requirement

7. Diagrammatic Representation of the Overall System

The figure below represents the complete architecture of our AI-based Phishing Email Detection and Generation System. It highlights the interaction between the user interface, the phishing email generator (GPT-2), and the phishing email detector (BERT).



- **User Interface (GUI)** using ttkbootstrap and tkinter
- **Phishing Email Generator Module** (GPT-2)
    - Takes user input or random prompts (sender, subject, message type)
    - Outputs a generated phishing-style email
- **Phishing Detector Module** (BERT)
    - Takes user-pasted or generated email text
    - Classifies as "Phishing" or "Legitimate"
- **Clipboard & Tab Integration**
    - Allows sending generated content to the detector
    - Provides real-time classification results